

Organizando os arquivos do projeto

Transcrição

Continuaremos implementando alguns detalhes, como o arredondamento dos cantos das imagens e a alteração de formatação das *labels*. Antes disso, porém, nos atentaremos a um ponto importante: nosso projeto ainda está no começo, porém já criamos alguns arquivos, e é recomendado organizá-los. Assim, conforme o projeto vai crescendo, adquirimos uma boa estrutura, com conteúdo facilmente identificável.

Optaremos pela organização por **funcionalidades**, ou features. Por ora temos apenas uma tela, a Home, e isso torna a organização bem mais simples. Com o lado direito do mouse, clicaremos em "Alura Viagens" na parte de diretórios, e em "New Group". Assim, criaremos uma pasta chamada "Model" com todas as nossas classes, e a arrastaremos para baixo de `ViewController.swift`.

A única classe que temos no projeto neste momento é `Viagem.swift`, portanto clicaremos nela e a arrastaremos para dentro de "Model". Daí em diante, basta criar e nomear as pastas de acordo com as funcionalidades que elas irão conter. No caso, criaremos uma pasta e a arrastaremos para que fique logo abaixo de "Model" (e **não dentro**). Nela, colocaremos `ViewController.swift` e `TableViewCell.swift`, e a nomearemos de "Home".

Em seguida, é interessante organizarmos o conteúdo destas pastas em subpastas denominadas "Controller" e "View". Então, criaremos as pastas em "Home", e na primeira colocaremos `ViewController.swift`, enquanto na outra incluiremos o arquivo `TableViewCell.swift`. Vamos criar uma última pasta ("ViagemDAO") para guardar o arquivo DAO, `ViagemDAO.swift`.

Voltaremos ao *storyboard* para trabalharmos a customização das *labels*. Os nomes dos destinos de viagens deverão ter:

- fonte Avenir Black
- tamanho 16
- cor RGB: #858585

Enquanto a quantidade de dias terá:

- fonte Avenir Medium
- tamanho 15
- cor RGB: #858585

E o preço:

- fonte Avenir Medium
- tamanho 15
- cor RGB: #858585

Com a célula selecionada, acessaremos o inspetor de atributos do lado direito do programa, e faremos as devidas alterações, assim como já fizemos antes. E para o arredondamento dos cantos das imagens, abriremos o `ViewController.swift` na pasta "Controller". No método `cellForRowAt`, repetiremos o procedimento feito as *Views* dos botões. Ou seja, incluiremos a linha `cell.imagemViagem.layer.cornerRadius = 10`, junto às demais.

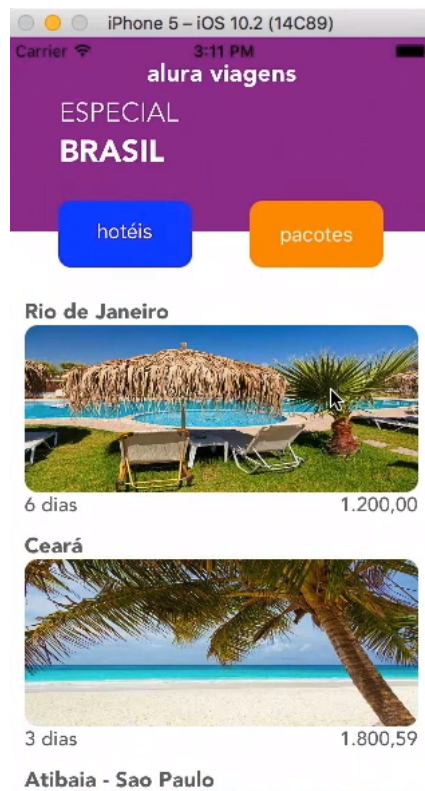
No entanto, em relação à imagem, ainda precisamos incluir um segundo atributo, que é o `masksToBounds`, uma máscara de recorte que faz a imagem ficar arredondada, passando como valor `true`.

```
cell.labelTitulo.text = viagemAtual.titulo
cell.labelQuantidadeDias.text = "\\(viagemAtual.quantidadeDeDias) dias"
cell.labelPreco.text = viagemAtual.preco
cell.imagemViagem.image = UIImage(named: viagemAtual.caminhoDaImagem)

cell.imagemViagem.layer.cornerRadius = 10
cell.imagemViagem.layer.masksToBounds = true
```

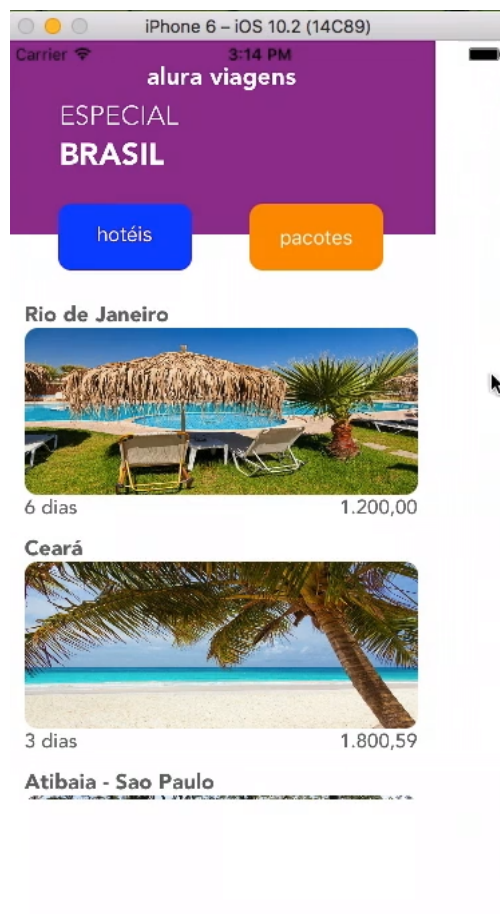
Vamos rodar o app para verificar o resultado destas alterações?

Nossa aplicação está com uma interface bem próxima àquela desejada:



Estamos trabalhando a todo instante com o simulador do iPhone 5, e sendo assim não encontraríamos problemas caso quiséssemos "buildar" o app neste dispositivo. Vamos fazer um teste e rodar no iPhone 6?

No menu superior do Xcode, clicaremos em "iPhone 5" para trocarmos para "iPhone 6", e então rodaremos o app novamente. Obteremos o seguinte resultado:



Isso acontece pois não configuramos os elementos do *storyboard*, ou *Interface Builder*, para que eles se reposicionem de acordo com o *device* em uso. Em resumo, da maneira em que está, a nossa aplicação funciona apenas em iPhone 5, o que é um problema. Veremos a seguir como resolver isso.